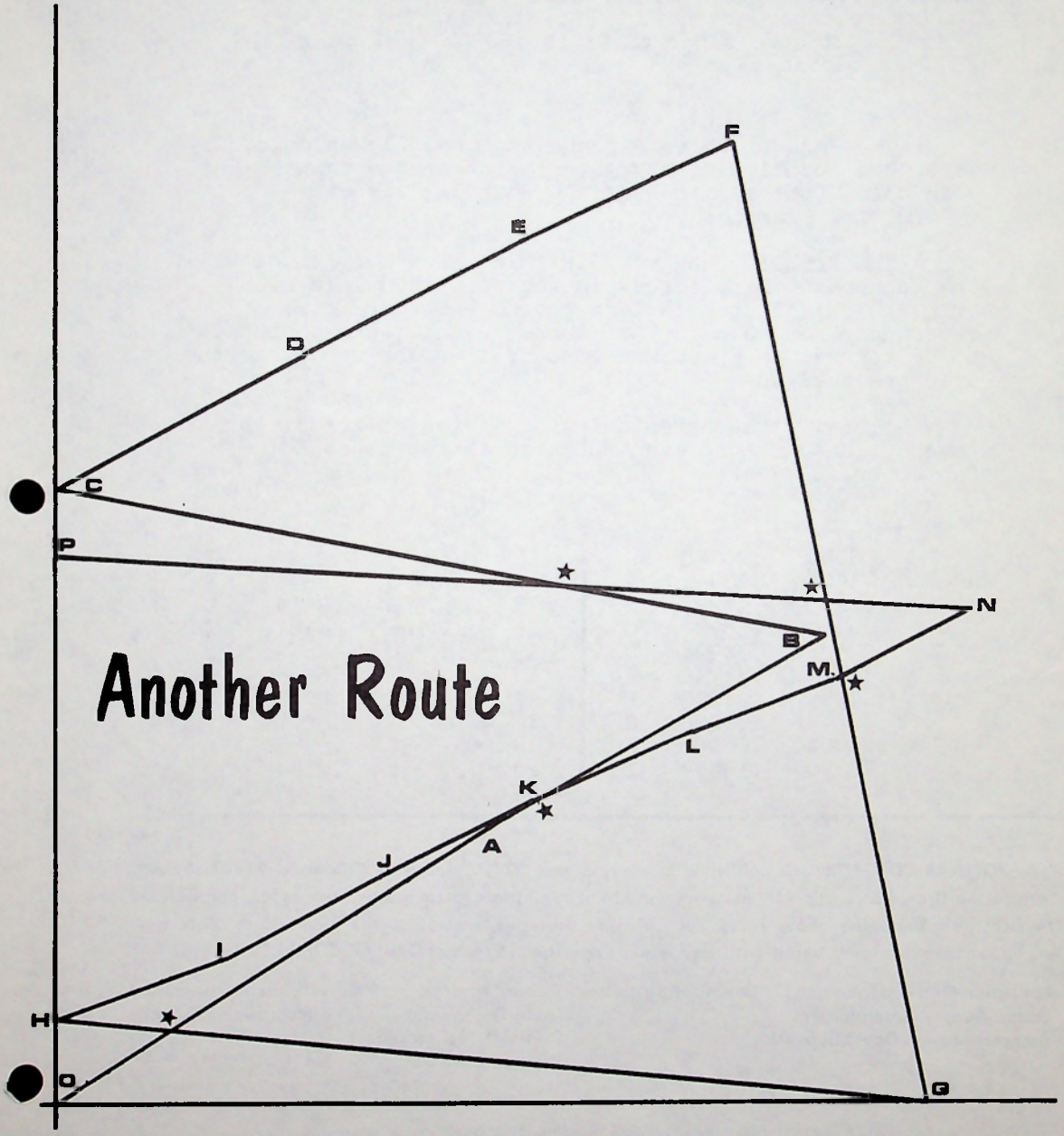
Popular Computing

BOX 272 CALABASAS, CA. 91302 VOL 2 NO 9

SEPTEMBER 74



Another Route

In the table shown below, the entries are the mantissas of the square root and cube root of successive integers, uniformly rounded to 5 decimal places. Each pair of roots furnishes the coordinates of a point in the unit square, and these points are connected in order, forming a trip, as shown on the cover for the first 16 points. (Although some segments appear collinear in the figure, they are not. For example, the slope of CD is .51925; the slope of DE is .50201; and the slope of EF is .48818).

The legs of this trip are thus readily calculated, as is the location of any given point on the trip. The 97th point (the terminus of the 96th leg) is at .84886, .59470, for example.

The Problem, however, is this: How many times will the connecting lines cross in the first 100 legs of the trip? (Within the 15 legs shown, there are five crossings, marked with the stars.) Note: every root is to be rounded to 5 decimal places, in order to define the trip precisely.

(This is the seventh in the series of Trips, which have appeared in the even numbered issues since No. 6, September, 1973.)

1	.00000	.00000	O	11	.31663	.22398	J
2	.41421	.25992	A	12	.46410	.28943	K
3	.73205	.44225	B	13	.60555	.35134	L
4	.00000	.58740	C	14	.74166	.41014	M
5	.23607	.70998	D	15	.87298	.46621	N
6	.44949	.81712	E	16	.00000	.51984	P
7	.64575	.91293	F	17	.12311	.57128	
8.	.82843	.00000	G	18	.24264	.62074	
9	.00000	.08008	H	19	.35890	.66840	
10	.16228	.15444	I	20	.47214	.71442	

POPULAR COMPUTING is published monthly at Box 272, Calabasas, California 91302. Subscription rate in the United States is $18 per year, or $15 if remittance accompanies the order. For Canada and Mexico, add $4 to the above rates. For all other countries, add $6 to the above rates. Back issues $2 each. Subscriptions may begin with any issue.

Publisher: Fred Gruenberger
Editor: Audrey Gruenberger
Associate editor: David Babcock

Contributing editors: Richard Andree, Irwin Greenwald, Daniel D. McCracken, William C. McGee

Advertising manager: Ken W. Sims
Art director: John G. Scott

Speaking of Languages

ROBERT TEAGUE

Some thought has already been given to the Fortran test deck proposed in PC17, and several contributions have already been received. Additional ideas concerning the design of the test will be welcomed. Also, the next issue will contain a revised and expanded table of results (given in PC15-14) to the three Fortran problems proposed in PC12-7. Readers have continued to work on the three problems and new output from different compilers has been obtained.

For this month's column, however, we will turn our attention to CØBØL as the language of interest. Although CØBØL was designed primarily as a data processing language, it does have some interesting computational and character-manipulation abilities.

Problem 1:

Using only a single, non-compound statement, find the absolute value of any value. The use of the IF statement is not allowed here, since it is considered for our purposes as a compound statement.

Problem 2:

Using only two non-compound statements, find the left-most non-zero digit of any data-item. Assume the data-item is defined without a computational sign and is stored in display form with two digits to the right of the point. Do not assume any size limitation to the value. (One purpose for such an exercise is to provide a starting point for printing the amount of a check in words, a process normally begun with the highest denomination.) Hint: a REDEFINES clause or two in the DATA DIVISIØN will be of help.

The earliest postmark with logically correct results for these puzzles as well as the most elegant solutions will appear in a future issue. Rush your results to

Speaking of Languages
POPULAR COMPUTING
Box 272
Calabasas, California 91302

Toward an 'A' Grade in computing

You have enrolled in an introductory course in computing, and would be pleased to emerge from it with a grade of A on your transcript. Unless it happens that the logic and symbolism of computing is beyond your understanding (and there are people for whom that is true), the guidelines to an A grade are simply stated.

We might consider first, though, the rationale of course grading. Why do we have grades at all? For decades, there have been attempts at eliminating grades, and there will be more. These experiments are almost always abandoned; the idea doesn't seem to work.

The first reason for having grades is tradition; that is, we've always had grading systems. Grades are a part of our culture; indeed, the phrase "He's a straight A student" is common outside of academic circles. Course grades are analogous to job salaries; it's the way the world assigns measures of progress and ability to a person's performance. Grades are a feedback mechanism; they measure something that provides guidance for future action.

Grades provide motivation. In a learning situation without grades, the "Why break my neck?" attitude becomes prevalent (and contagious).

Grades allow us to standardize. Although the measure is far from perfect (grades are assigned by people, not by computers), we are furnished with a means of comparing work. This principle becomes painfully obvious when a student tries to transfer credits from one school to another.

It is probable that, as you start work in learning computing, your interest is somewhat casual; that is, you have no intention of becoming a computer professional, or even of concentrating on the use of computers. The trouble is that at this point you can't predict the future that well. It is not at all uncommon for your computing course to become very important to you in later activities and in your career. As a teacher of computing, I am continually being called for reference information about former students who cite their computing course as a milestone among the things they've been exposed to. When the call is from a personnel manager, I'd like to be able to say "How has your firm managed so far without him?" I can sometimes ask that question for A students, but I can deal only in platitudes and banalities for the student to whom I gave a C.

So I operate on the theory that grades are necessary and that I'll be required to furnish one for you. I might be able to do this based only on my opinion of you, but it is helpful to have some numerical evidence. For that purpose, we have exams during the course. Actually, there are other, more compelling, reasons for exams and you should be aware of them:

1. Exams provide feedback to you. There is a tremendous amount of detail and peripheral information in a computing course. You cannot be expected to absorb all of it, nor should you. An exam that rests on factual information that you memorize and regurgitate would not measure your knowledge of computing very well. It is one of your tasks to discern the essence of the subject, and a good exam should let you know what it is that I think is important.

2. Exams also provide feedback to me. They tell me what information I have succeeded in getting you to learn, and—more importantly—what it is I thought I had taught but evidently hadn't. I need feedback as much as you do.

3. An exam—in order not to waste time—should allow me to teach you something. If nothing else, it should teach you how to take an exam, and that's something you'll be doing all your life. But I might also be able to teach you something about computing, perhaps by presenting you with a new view of some aspect of the subject.

4. The fourth reason for exams is the one we started with; namely, as a means of ranking you in relation to your classmates. This relation has something to do with "grading on a curve," which is a subject that my students seem to understand a great deal better than I do. We can agree on this: if an exam counts 100 points, and the greatest number that any student gets on that exam is 90, then the exam really counted only 90 points.

As you might notice, I go on the assumption that you are here to learn; that learning takes work (especially learning computing); and that grades should correlate with the amount of your learning. I want you to get a good grade, but the burden of proof is on you.

Getting back to the subject of exams, it has been my experience that few students have ever been told how to take an exam. The rules are simple and obvious:

1. *Do the easy ones first.* If you number your answers to correspond to the numbers of the questions (would you believe that some students don't do that?), then I'll find the answers, and I don't care what order they were written in. If you get out of the way the ones that you find easy, you'll have time at the end of the period to work on the hard ones. It really makes no difference, logically, but it is psychologically helpful that way.

2. *Write in English.* English is the only language I can read. If it's not in English, I can't read it, and if I can't read it, it's wrong. This principle unfortunately includes the subject of spelling. Spelling isn't really a topic in computing, and I'll try to make out what you write, but you should try to spell correctly, particularly words like "computer." I can think of no single subject that is more important to you that English communication. Without it, you will be of little use to anyone, no matter how capable you become technically; with it, you will have a tool that can, to some extent, mask your deficiencies in technical areas.

3. *Something is better than nothing.* If you leave a question out, or make a vague pass at it, I have no choice but to grade it zero. If you write something—anything—it may still be worth zero, but it is harder for me to assign that value.

4. *Show your work.* You've heard that one many times before, so I won't dwell on it.

5. *Give the man what he wants.* Solve the problem that is given to you, not some new problem that you happen to like better. Try to do it the way it was done in class; some other way may please you more, but why make it difficult for the person who grades it? Put your name on your paper *before* the exam begins, and in the place where your instructor prefers to have it. Think of it this way: you want him in a mellow mood as he grades your paper.

6. *Make up a simpler problem first.* This is the hardest idea to put across, and yet the most powerful tool for problem solving (which is what an exam is) that I know. You are faced with a problem, and you don't know where to begin (all good problems have that characteristic). Invent a new problem which is simpler and for which you know the answer. Ask yourself how you did that problem, and then generalize the result to the original problem. Let me illustrate:

Suppose there are 27 volumes on a shelf, and they are consecutively numbered from 43 through 69 (quick—does that check?). What is the number of the volume in the middle? Suppose you don't know how to do that; it involves subtracting, and dividing by two and then adding, or something. I'll make up a simpler problem. There are three books on the shelf, and they're numbered 35, 36, and 37. I can readily see the answer to this new problem: it is 36. Now, how can I get it from the 35 and 37? Let's see: the difference is two; divide that by two; add the result to the lower number. Just to check, I'll try it again with five books, which I can also readily picture. It seems to work, so I'll use it on my original big problem. Try this simple idea out on the next problem situation you meet and see if it doesn't help. It has solved many a computing problem for me. Come to think of it, it has nothing to do with computing problems; it's a general principle.

Despite your best efforts, and with the best of intentions, you may find computing a difficult subject to learn. Part of the difficulty lies in the fact that you are essentially learning a new language which is being taught *in* that language. There is some small comfort in the knowledge that a great many people—some of them much less bright than you—have already learned it, and it wasn't easy for them, either.

Soon after the start of an introductory course in computing, students waylay the instructor outside of class with the complaint, "I don't seem to understand this subject." That may be a fair complaint. On the other hand, it's not at all fair if what it says is, "I don't seem to understand this subject without working." Has someone told you that you could learn computing without doing some real work?

Rather than go through the ensuing dialog with each student who has this complaint, let me give you a copy of the script . . .

1. Have you tried? (Just for example, if you have not handed in the homework assignments, the answer is "no.")

2. Have you asked questions in class? The canonical response to this is always, "I don't understand enough to be able to ask a question." That isn't very sensible. The really stupid question is the one you don't ask, at the time when something isn't clear. If your questions did become obtrusive, I could always cut you off, and that seldom happens.

There *is* one question you can always ask: "Mr. Jones, would you mind going over that point once more?" Try it sometime, and notice the grateful looks you get from the more timid students. Oh, that's it: you're timid? Well, write me a note, or send up a rocket, but don't just sit there if something is getting past you. Things move pretty fast in a computing course, and if you fall behind, you may stay behind.

3. Have you studied? This doesn't mean scanning the text or your notes; it means studying. Have you examined carefully the routines presented to you? Can you account for every character in every instruction? Can you rewrite the routines without looking at your notes or the text? Can you account for every red-pencil mark on your exam papers and your homework? If any exam questions were to be repeated on a subsequent exam, would you have the answers down cold the second time?

4. Can you explain why *you* think you're confused?

5. In a computing course, there is one—and really only one—road to understanding, and that's by actually computing. We have computing power available to you—use it. It is not just coincidence that those students who plunge in and do some computing catch on to the subject much faster.

I have been trying to explain my assumption that you and I have a similar goal; namely, to have you learn something about computing. It may be, with the best of intentions on the part of both of us, that we do not hit it off; that some friction develops between us. In that case, the following quotation from Walt Whitman might be of interest:

> "Have you learn'd lessons only of those who admired you, and were tender with you, and stood aside for you? Have you not learn'd great lessons from those who reject you, and brace themselves against you? or who treat you with contempt, or dispute the passage with you?"

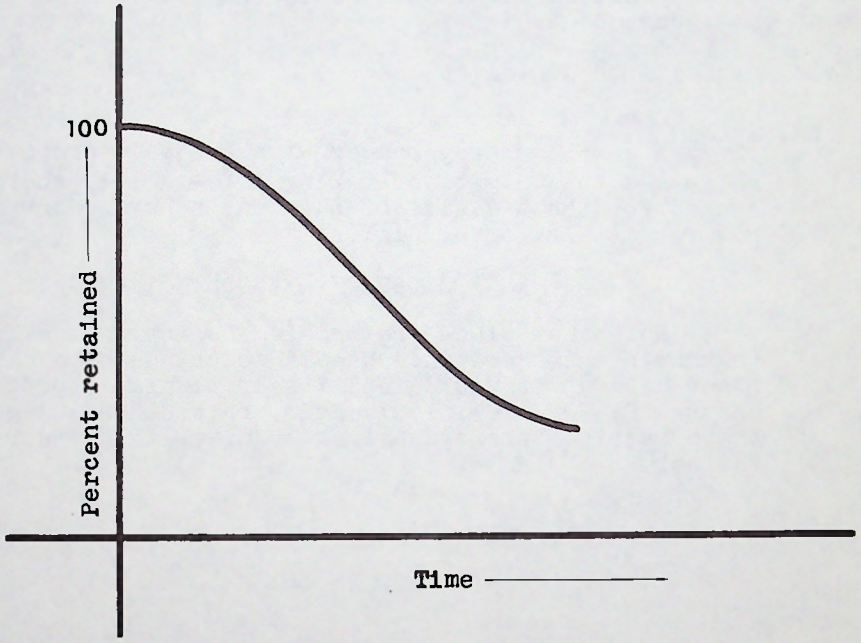
A common question that arises when a student discovers that he is doing poorly is "Is there anything I can do for extra credit?" No, there isn't. Stop to think about it. If there is any extra credit mechanism set up, which students would be sure to do all of it promptly and well?—the top students. Thus, the mechanism would surely operate to *lower* the relative position of the poor student.

A gag phrase in computing is "Do you want it right or do you want it right away?" All too often, the one who seeks a computer solution wants it right away, even if it's not right. Eventually he'll want it right, of course, which leads to another catch phrase: "Why is there never time to do it right, but always time to do it over?" Your homework assignments are miniature simulations of real-life computer situations. It's really easier to get them right before they're due.

All work in computing should be done with McDougall's Law in mind: If anything can go wrong, it will, and it will happen to you, and at the worst possible time ("McDougall's bread always falls jelly-side down."). The naive programmer assumes that everything will be as it should be; that machines, and operators, and key-punchers, and other programmers are somehow perfect. It ain't so.

There is no a priori reason to believe that your entire class could not be A students, or, for that matter, F students. Past experience indicates, however, that differences in native ability, motivation and drive, and that peculiar sense of puzzle-solving that characterizes computing, all operate to spread students out. To a certain extent, an instructor seeks to identify the differences between students. In any event, at the end of a semester, total grade points usually show a wide range. You may examine one of these distributions from a previous semester if you wish.

The figure below shows the "forgetting curve." Whatever is learned tends to evaporate with time, and the time scale shown is measured in *hours*. The point is this: if you do your studying, and review, and homework as soon as possible after each class, it is not only more sensible (that is, it avoids the human tendency to procrastinate), but it is actually much more efficient and easier (not to discount the comfortable and smug feeling of having it out of the way). In computing, you will be hit with new concepts in almost every class period, as well as new words and notation. In as little as four hours after a class, some of this material is already hazy, even when it was clear at the time you heard it.



A large collection of nonsense never adds up to good sense. Try to seek neat, elegant, and short solutions. Naturally, it is a question of judgment as to what level of detail (on a flowchart, say) is sufficient, and what level begins to border on coding. Some students express great surprise that this is so, particularly when their judgment doesn't agree with mine. There should be no surprise; if we could define matters of judgment rigorously, we'd write a computer program for it, and our industry would need no more programmers.

Any written work that requires a personal lecture of explanation to go with it is not very good work.

A great many people have already learned the rudiments of computing. You can, too. Don't be misled by the fact that the whole subject will be terribly confusing for about the first month or so. It *does* clear up, provided you put in the necessary time and effort. And most students who take the final exam testify that the study of computing was one of the most interesting courses they ever took.

A Number that Extends Infinitely Far to the Left

The number

A = 4106619977392256259918212890625

has the property that its square has, for its 31 low order digits, the number A itself. Sanford Greenfarb, Fort Bragg, North Carolina, points out that the number can be extended infinitely far to the left. Given K digits of A, the (K+1)st digit will be the (K+1)st digit of the square of A.

Problem: extend A to 100-digit length.

A complementary number with the same property can be formed to N digits by taking $10^N - X + 1$, where X stands for the N digits of A. The other number thus has for its low order digits:

5893380022607743740081787109376.

Note that since the problem is intrinsically decimal (although there are undoubtedly solutions obtainable in other bases), it would lend itself to simple programming on the IBM 702, 705, 1401, 1620, or the RCA 301--but all these machines are endangered species, if not already extinct.

"It seems that the machine, when cursed for being too slow or too small, may often in justification demand in turn that the user do some more thinking."

—Hao Wang

Reciprocals

7	.142857142857142857142857142857142857142857142857
13	.076923076923076923076923076923076923076923076923
17	.058823529411764705882352941176470588235294117647058823
19	.052631578947368421052631578947368421052631578947368421
21	.047619047619047619047619047619047619047619047619047619
23	.043478260869565217391304347826086956521739130434782609
29	.034482758620689655172413793103448275862068965517241379
31	.032258064516129032258064516129032258064516129032258065
41	.024390243902439024390243902439024390243902439024390244
43	.023255813953488372093023255813953488372093023255813953
47	.021276595744680851063829787234042553191489361702127660
53	.018867924528301886792452830188679245283018867924528302
59	.016949152542372881355932203389830508474576271186440678
61	.016393442622950819672131147540983606557377049180327869
67	.014925373134328358208955223880597014925373134328358209
71	.014084507042253521126760563380281690140845070422535211
79	.012658227848101265822784810126582278481012658227848101
83	.012048192771084337349397590361445783132530120481927711
89	.011235955056179775280898876404494382022471910112359551
97	.010309278350515463917525773195876288659793814432989691
103	.009708737864077669902912621359223300970873786407766990
107	.009345794392523364485981308411214953271028037383177570
109	.009174311926605504587155963302752293577981651376146789
113	.008849557522123893805309734513274336283185840707964601 769911504424778761061946902654867256637168141592
127	.007874015748031496062992125984251968503937007874015748
131	.007633587786259541984732824427480916030534351145038167 938931297709923664122137404580152671755725190839
139	.007194244604316546762589928057553956834532374100719424
149	.006711409395973154362416107382550335570469798657718120 805369127516778523489932885906040268456375838926
151	.006622516556291390728476821192052980132450331125827815
157	.006369426751592356687898089171974522929936305732484076 433121019108280254777070063694267515923566878 98
163	.006134969325153374233128834355828220858895705521472392 638036809815950920245398773006134969325153374233
167	.005988023952095808382335329341317365269461077844311377 245508982035928143712574850299401197604790419 16
173	.005780346820809248554913294797687861271676300578034682
179	.005586592178770949720670391061452513966480446927374301 6759776536312849**1620**111731843575418994413407821 2
181	.005524861878453038674033149171270718232044198895027624 309392265193370165745856353591160220994475138121
191	.005235602094240837696335078534031413612565445026178010 471204188481675392670157068062827225130890052356
193	.005181347150259067357512953367875647668393782383419689 119170984455958549222797927461139896373056994818
197	.005076142131979695431472081218274111675126903553299492 385786802030456852791878172588832487309644670050

The reciprocals of certain primes. The ones carried to 102 decimal places are those having long periods of repetition. The ones expressed to 54 decimals are rounded in the 54th place.

1	00	.5
2	00	.25
3	00	.125
4	01	.625
5	01	.3125
6	01	.15625
7	02	.78125
8	02	.390625
9	02	.1953125
10	03	.9765625
11	03	.48828125
12	03	.244140625
13	03	.1220703125
14	04	.6103515625
15	04	.30517578125
16	04	.152587890625
17	05	.762939453125
18	05	.3814697265625
19	05	.19073486328125
20	06	.95367431640625
21	06	.476837158203125
22	06	.2384185791015625
23	06	.11920928955078125
24	07	.5960464477539062 5
25	07	.29802322387695312 5
26	07	.14901161193847656 25
27	08	.74505805969238281 25
28	08	.37252902984619140 625
29	08	.18626451492309570 3125
30	09	.93132257461547851 5625
31	09	.46566128730773925 78125
32	09	.23283064365386962 890625
33	09	.11641532182693481 4453125
34	10	.58207660913467407 2265625
35	10	.29103830456733703 61328125
36	10	.14551915228366851 806640625
37	11	.72759576141834259 033203125
38	11	.36379788070917129 5166015625
39	11	.18189894035458564 75830078125
40	12	.90949470177292823 79150390625
41	12	.45474735088646411 895751953125
42	12	.22737367544323205 9478759765625
43	12	.11368683772161602 97393798828125
44	13	.56843418860808014 86968994140625
45	13	.28421709430404007 43484497070312 5
46	13	.14210854715202003 71742248535156 25
47	14	.71054273576010018 587112426757812 5
48	14	.35527136788005009 29355621337890625
49	14	.17763568394002504 64677810668945312 5
50	15	.88817841970012523 23389053344726562 5
51	15	.44408920985006261 61694526672363281 25
52	15	.22204460492503130 80847263336181640625
53	15	.11102230246251565 40423631668090820312 5
54	16	.55511151231257827 02118158340454101562 5
55	16	.27755575615628913 51059079170227050781 25

NEGATIVE POWERS OF 2

(or, reciprocals of powers of 2)

The first column is the power. The second column indicates the number of zeros that should be placed after the true decimal point. Thus,

$2^{-10} = .0009765625$

56	16	.13877787807814456755295395851135253906 25
57	17	.69388939039072283776476979255676269531 25
58	17	.34694469519536141888238489627838134765625
59	17	.17347234759768070944119244813919067382812 5
60	18	.86736173798840354720596224069595336914062 5
61	18	.43368086899420177360298112034797668457031 25
62	18	.21684043449710088680149056017398834228515625
63	18	.10842021724855044340074528008699417114257812 5
64	19	.54210108624275221700372640043497085571289062 5
65	19	.27105054312137610850186320021748542785644531 25
66	19	.13552527156068805425093160010874271392822265625
67	20	.67762635780344027125465800054371356964111328125
68	20	.33881317890172013562732900027185678482055664062 5
69	20	.16940658945086006781366450013592839241027832031 25
70	21	.84703294725430033906832250067964196205139160156 25
71	21	.42351647362715016953416125033982098102569580078125
72	21	.21175823681357508476708062516991049051284790039062 5
73	21	.10587911840678754238354031258495524525642395019531 25
74	22	.52939559203393771191770156292477622628211975097656 25
75	22	.26469779601696885595885078146238811314105987548828125
76	22	.13234889800848442797942539073119405657052993774414062 5
77	23	.66174449004242213989712695365597028285264968872070312 5
78	23	.33087224502121106994856347682798514142632484436035156 25
79	23	.16543612251060553497428173841399257071316242218017578125
80	24	.82718061255302767487140869206996285356581211090087890625
81	24	.41359030627651383743570434603498142678290605545043945312 5
82	24	.20679515313825691871785217301749071339145302772521972656 25
83	24	.10339757656912845935892608650874535669572651386260986328125
84	25	.51698788284564229679463043254372678347863256931304931640625
85	25	.25849394142282114839731521627186339173931628465652465820312 5

NEGATIVE POWERS OF 2 Continued

Artificial Irrationals

PROBLEM 60

By definition, a fractional sequence that does not repeat and does not end, is an irrational number. In decimal notation, the sequences that have been presented as the "N-Series" in POPULAR COMPUTING are irrational (and most of them are also transcendental). The following sequences in binary notation are contrived to fit the definition of irrational:

A .1001110000111110000001111111100000000111111111...
(One 1, two 0s, three 1s, four 0s, five 1s, six 0s,...)

B .1011001110001111000011111000001111110000001111111...
(One 1, one 0, two 1s, two 0s, three 1s, three 0s, four 1s, four 0s, five 1s, five 0s, six 1s, six 0s,...)

C .1110101000101000101000100000101000001000101000100...
(1 wherever the position number after the binary point is prime; that is, 1 in position 1, 2, 3, 5, 7, 11,...)

D .1011011101111011111011111101111111101111111110111111111...
(One 1, 0, two 1s, 0, three 1s, 0, four 1s, 0, five 1s,...)

E .1010010001000010000010000001000000010000000010000000001...
(1, 0, 1, two 0s, 1, three 0s, 1, four 0s, 1, five 0s,...)

F .1011100111011110100000011010111000001111011111111 0...
(1s where the corresponding decimal digit of pi is odd, 0s where the corresponding decimal digit of pi is even)

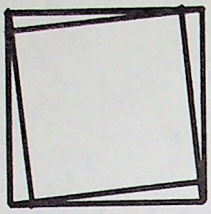
G .1011000111110000000011111111111111000000000000000000000000...
(Alternate 1s and 0s, taking each the number of times given by the Fibonacci sequence (1, 1, 2, 3, 5, 8, 13,...))

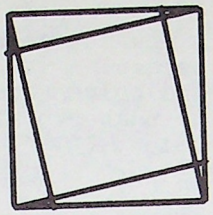
H .1010001001000010001000000100001000000100000100000001...
(Pattern of 0s between the 1s follows the scheme 1, 3, 2, 4, 3, 5, 4, 6, 5, 7, 6, 8, 7, 9, 8, 10,...)

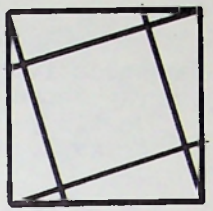
J .0100001000000000100000000000000001000000000000000000000000...
(0s separate the 1s in a pattern of squares; that is, the number of 0s is 1, 4, 9, 16, 25, 36,...)

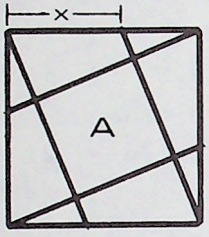
The Problem is:
What are the decimal equivalents of these binary irrationals? Each decimal value should be calculated correct to 20 decimal places.

C





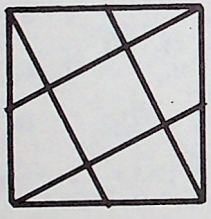


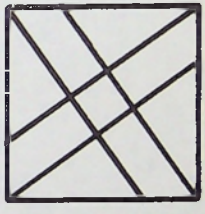


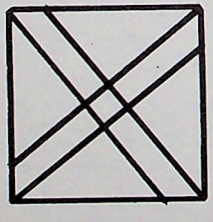
If distances X are marked off on each side of a unit square and those points are connected to the opposite corners, a new square, A, is formed. When X is close to zero, as in diagram B, the area A is close to zero; when X is close to 1, as in diagram C, the area A is close to 1. But the area A does not change in proportion to X, but rather as shown in diagram D.

Problem: Find the equation of the curve that relates A and X.

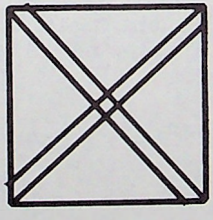
The above is not a computing problem, but rather a problem in analytic geometry. Suppose now that the distances X on the sides of the unit square are not equal, as in diagram E. Given the four values of X (e.g., .8, .6, .4, and .2), what is the area A? Write a Fortran subroutine for which the arguments are the four X values (in specified order, and all less than one) and the output of the subroutine is the area A.



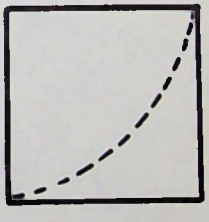


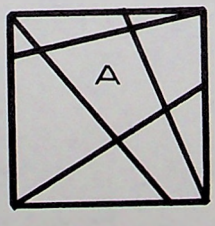


B



D





PROBLEM 61

Spiral Transformation

An array is stored in row and column order in consecutive locations in storage. Thus, for the 5 x 5 array shown at G, the row and column designations are given in parentheses and the number of the consecutive storage location is given below.

G

(1,1) 1	(1,2) 2	(1,3) 3	(1,4) 4	(1,5) 5
(2,1) 6	(2,2) 7	(2,3) 8	(2,4) 9	(2,5) 10
(3,1) 11	(3,2) 12	(3,3) 13	(3,4) 14	(3,5) 15
(4,1) 16	(4,2) 17	(4,3) 18	(4,4) 19	(4,5) 20
(5,1) 21	(5,2) 22	(5,3) 23	(5,4) 24	(5,5) 25

PROBLEM 62

The numbers in the array are to be moved so that they appear in the array in spiral fashion, as shown in figure H.

1. Flowchart the logic for performing the spiral transformation for any size array larger than 2 x 2.

2. Write a program in Fortran for a subroutine, whose arguments in the calling sequence are

NAME1, NAME2, ORDER

where NAME1 is the name of the array to be moved; NAME2 is the name of the new array; and ORDER is the size of the array (e.g., 5).

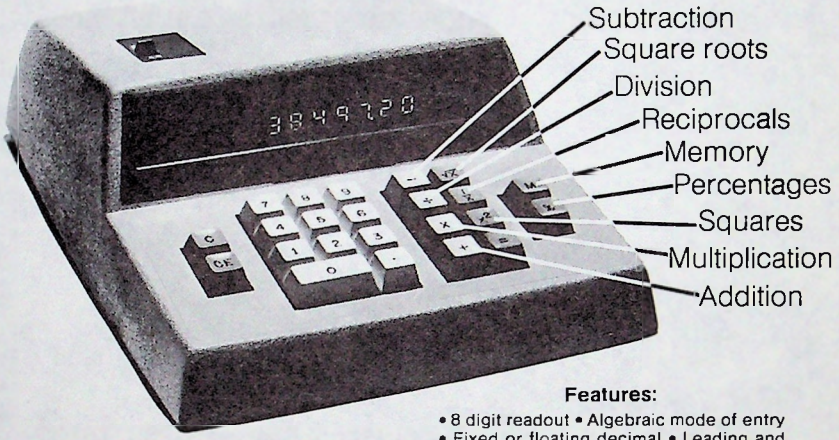
A Fortran Transformation

H

1	2	3	4	5
16	17	18	19	6
15	24	25	20	7
14	23	22	21	8
13	12	11	10	9

MITS Presents

The new 908DM, Desk-Top Calculator.



Features:

• 8 digit readout • Algebraic mode of entry • Fixed or floating decimal • Leading and trailing zero suppression • Chain and mixed operations

✱ Plus the option of programmability.

Full Operation Memory

Memory may be used as:

1. A constant
2. A temporary storage register
3. An accumulator

Indicators:

• True credit balance sign • Overflow

***Prices: 908DM**

Kit. . **$129.95** Assembled. . . **$149.95**

Size: 8-1/2" x 12" x 3-1/4"

✱Programmer

To be used with the MITS 816, 1440, or the new **908DM**, desk calculators.

1. Provides 256 programming steps. (With option of expandability to 512 steps.)
2. Stores up to 64 separate programs. Size: 8-1/2" x 12" x 3-1/4"

Instructions:

A. "If Neg," B. "Go To," C. "Return," D. "Remember," E. 2 Run modes of operation

*Programmer Kit. . . . . **$199.95** Assembled. **$299.95**

*Combination 908DM and Programmer Kit. . . . . **$299.95** Assembled. **$399.95**

Warranty: Kit: 90 days on parts, Assembled: 2 years on parts and labor.

*Prices subject to change without notice. Available from your local Olson Electronics Dealer

MITS INC.

"Creative Electronics"

6328 Linn, N.E., Albuquerque, New Mexico 87108
505/265-7553 Telex Number 660401

☐ Enclosed is a Check for $ _____
or ☐ BankAmericard # _____
or ☐ Master Charge # _____
Credit Card Expiration Date _____ ☐Kit ☐Assembled
Include $5.00 for Postage and Handling
☐Model 908DM ☐Programmer ☐908DM & Programmer
☐ Please Send Information on Entire MITS Line.
NAME _____
ADDRESS _____
CITY _____
STATE & ZIP _____
MITS/ 6328 Linn, N.E., Albuquerque, New Mexico 87108 505/265-7553 Telex # 660401

CW-W-74

Log 18	1.25527250510330606980379470123472364516844760984350 0
Ln 18	2.89037175789616469220772259530322797737048125000575 4
$\sqrt{18}$	4.24264068711928514640506617262909423570901562613084 4
$\sqrt[3]{18}$	2.62074139420889660714166128044199627023942764572363 2
$\sqrt[5]{18}$	1.78260245796600335549488747214008661063589569472799 6
$\sqrt[7]{18}$	1.51120939050940313569103411141329150954930881965601 2
$\sqrt[10]{18}$	1.33514136254031294073673878070317172035361589847808 2
$\sqrt[100]{18}$	1.02932548375428497228312546690815489099402875923918 2
e^{18}	65659969.13733051113878650325906003356921635578618681 9491493276417998571731739128559146722553833 7
π^{18}	888582403.07126338067024359596537231630496097157585025 1626791243031465585975915698500056878958 56
$\tan^{-1}$ 18	1.51529782154917978367412354354751423165878135462816 4 8813479133852763158599247918745255114462680170079 96
18^{100}	33670573242751689858746062777200469754260529531607724 7452351004740460372771448563031143507354757009822674 6 1869740537956374937 6

"It is obvious that the most ignorant savages can use computers today and still be ignorant and still be savages, and this may be our most important single problem."
—Clarence B. Poland III

Back issues are still available.

JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC		
			1	2	3	4	5	6	7	8	9	Vol. 1	1973
10	11	12	13	14	15	16	17	18	19	20	21	Vol. 2	1974
22	23	24	25	26	27	28	29	30	31	32	33	Vol. 3	1975